19 | 第三方工具库：最常用的第三方工具库有哪些？

2021-07-08 王沛

《React Hooks 核心原理与实战》 课程介绍 >

讲述：王沛
时长 13:03 大小 11.96M

你好，我是王沛。今天我们来聊聊在 React 开发过程中有哪些常用的第三方工具库。

如今的软件开发已经离不开开源社区提供的各类工具库了，合理的使用它们，不仅可以帮助节省重复的开发时间。而且通常来说，流行的开源库具有更充分的测试和更高的质量。因此，在 React 的开发过程中，知道 React 生态圈有哪些常用的工具库，了解它们能解决什么问题，是非常有必要的，我们要学会站在前人的肩膀上去成就自己嘛。

那么今天这节课，我会和你分享我自己最常用的四个工具库，分别是 lodash、Material UI、Ant.Design 和 react-use。我会通过一些例子来演示它们的部分功能和用法，让你对

它们有一个总体的了解。

lodash

首先想和你分享的就是 lodash。在我看来，它是每一个前端开发者都应该熟练掌握的。

lodash 提供了非常多的工具函数，就像一把强大的瑞士军刀，能够满足各种前端开发需求，从而可以大大提高开发的效率。而且，lodash 不仅能用在浏览器端，也可以用在 NodeJS 的开发中。所以说，掌握了 lodash ，也能提升 NodeJS 的开发效率。

下面我会通过三个短小的例子，来让你体会 lodash 带来的方便，这样我就不用给你解释什么叫惊喜了。

使用 keyBy，将数组快速转换成对象

lodash 提供了大量操作数据结构的工具函数，其中 keyBy 是我最常用的方法之一。通常来说，我们从 API 会拿到数组存储的一组数据，为了对数据进行操作，我们一般需要根据数据的某个属性，快速找到数组元素，而 keyBy 可以帮助我们快速做到这一点。

比如，对于如下的数据结构：

复制代码

```
const data = [
  {
    id: "1",
    name: "Kennedy",
    job: "Chief Mobility Orchestrator",
    city: "North Alec",
  },
  {
    id: "2",
    name: "Lucius",
    job: "Internal Research Manager",
    city: "Littleland",
  },
  {
    id: "3",
    name: "Carlos",
    job: "Lead Configuration Analyst",
    city: "South Lillian",
  },
]
```

如果要将其变成一个以 name 为 key 的 map 结构，那么只要用如下的代码就能完成。

复制代码

```
import _ from 'lodash';
const byName = _.keyBy(data, 'name');
```

可以看到，一行代码就实现了这个转换，而且语义上也非常清楚。

使用 debounce 函数，实现输入防抖

在带有搜索框的页面里，我们通常会将搜索框的用户输入，保存为 searchKey 这样一个 state，当 searchKey 发生变化的时候去刷新搜索结果。但是，如果每次用户敲击一下键盘都去刷新搜索，其实是没有必要的，因为用户很可能还没有输入完关键字。而且，多余的搜索，还有可能会引起输入过程的卡顿。

为了解决这个问题，我们通常会使用防抖机制，也就是只有在用户停止输入后一个很短的时间内才进行搜索，比如说 300 毫秒，以保证更好的用户体验。而 lodash 就专门提供了 debouce 这样的一个函数。

它的用法如下：

复制代码

```
import React, { useMemo } from "react";
import _ from "lodash";
import { useSearchParam } from "react-use";


function SearchBox({ data }) {
  const searchKey = useSearchParam("key") || "";
  const filtered = useMemo(() => {
    return data.filter((item) =>
      item.title.toLowerCase().includes(searchKey.toLowerCase()),
    );
  }, [searchKey, data]);


  // 使用 _.debounce 处理用户输入，300ms 后才执行
  const handleSearch = _.debounce((evt) => {
    window.history.pushState(
```

```
      {},
      "",
      `${window.location.pathname}?key=${evt.target.value}`,
    );
  }, 300);
  return (
    <div className="08-filter-list">
      <h2>Movies (Debounced Search)</h2>
      <input
        defaultValue={searchKey}
        placeholder="Search..."
        onChange={handleSearch}
      />
      <ul style={{ marginTop: 20 }}>
        {filtered.map((item) => (
          <li key={item.id}>{item.title}</li>
        ))}
      </ul>
    </div>
  );
```

在这里我们使用了 lodash 的 debounce 方法封装了 handleSearch 这个事件处理函数，这样的话，就只有在停止输入 300 毫秒之后，才真正去执行搜索这个动作。

实际的效果你也可以参考在线的示例：https://codesandbox.io/s/react-hooks-course-20vzg 。

使用 _.template，实现简单的模板引擎

lodash 提供了一个简单的模板引擎，可以非常方便地让你去实现一些复杂的字符串生成。

比如下面就是一个例子：

复制代码

```
var compiled = _.template('<% _.forEach(users, function(user) { %><li><%- user
compiled({ 'users': ['fred', 'barney'] });
// => '<li>fred</li><li>barney</li>'
```

可以看到，_.template 通过一定的开始和结束标记，让你能够在其中使用条件判断、循环等功能。这在我们需要去生成类似 HTML、Markdown 等大段的文档时，就会非常有用。此外，将它作为一些样本代码的生成器，也是非常好用的。

当然，lodash 的功能远远不止于此，这里介绍的三个例子只是让你先对它有一个直观的了解和感受，更多的惊喜等待你自己去发掘。详细的 API 你可以去参考官方文档。

UI 库：Ant Design 和 Material UI

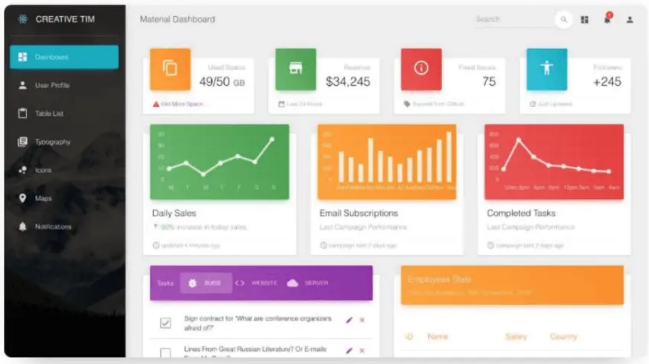
UI 组件库是前端开发必不可少的一部分，因此在开始开发 React 时，要做的第一件事大概就是选择 UI 库了。这里我主要介绍两个常用的 UI 库。

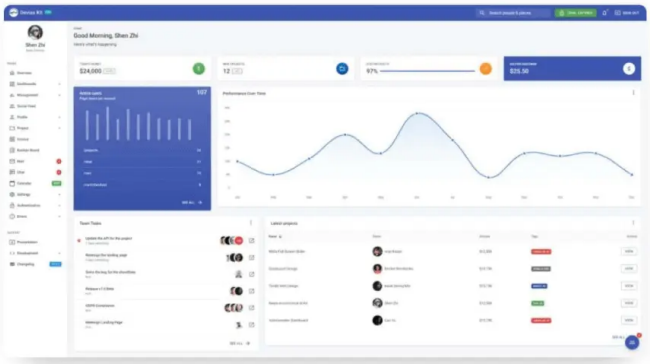
> 一个是 Material UI（以下简称 MUI）。它采用了 Google 的 Material Design 的设计语言，提供了丰富的 UI 组件库。
>
> 另一个则是国内蚂蚁金服开源的 Ant.Design（以下简称 AntD）组件库，主打企业级应用的场景。

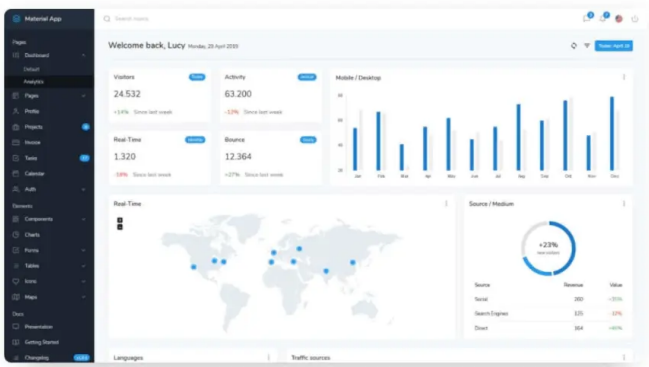
Material UI

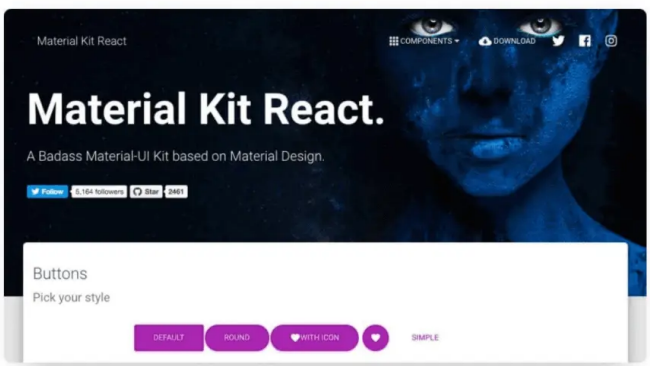
虽然 MUI 是基于 Google 的 Material Design，但是它提供了非常灵活的主题定制功能，这也是区别于 Ant Design 的一大特色。这就意味着基于 MUI 有一个庞大的主题市场，你可以从中选择自己需要的主题。

下图就是 MUI 官网提供的一些主题：









MUI 实现了一个强大的主题系统，主要原因就在于使用了 React Context 和 Styled Component 来实现 JavaScript 控制的动态主题系统，这也是它的基本原理。

比如下面这段代码，就展示了该如何定制一个主题：

复制代码

```
import React from 'react';
import styled, { ThemeProvider } from 'styled-components';
import { createMuiTheme } from '@material-ui/core/styles';
import { palette, spacing, typography } from '@material-ui/system';
// 使用 JavaScript 定制样式
const Box = styled.div`${palette}${spacing}${typography}`;
const theme = createMuiTheme();
export default function Demo() {
  return (
    <ThemeProvider theme={theme}>
      <Box
        color="primary.main"
        bgcolor="background.paper"
        fontFamily="h6.fontFamily"
        p={{ xs: 2, sm: 3, md: 4 }}
      >
        @material-ui/system
      </Box>
    </ThemeProvider>
  );
}
```

这样，通过在你的组件中组合主题中定义的各种样式的规则，比如间距、字体、颜色等等，让你的组件就能够去适应各种不同的主题，从而保持 UI 样式的一致性。因此，如果你的应用对个性化的主题有比较高的需求，那么 Material UI 是不错的选择。

虽然 MUI 很酷炫，但它也有一个很大的缺点：**组件库中缺少一些高级的功能**。比如复杂的 Table、Tree 等，通常都需要依赖第三方来实现，而不是 MUI 自身就包含的。我们都知道，Table 和 Tree 是企业级应用最常用的组件。如果要依赖第三方，那质量和稳定性就参差不齐了。

那你可能就会问了，我到底该使用 MUI 还是 AntD 呢？我的一个标准是：要看你的应用是 To C 还是 To B。

如果是 To C，就用 MUI，因为个性化的外观很重要。而如果是 To B，那就建议使用接下来要介绍的 AntD，因为它就是面向企业级应用设计的，风格以极简为主，组件库更丰富，功能也更强大。

Ant Design

Ant Design 在国内有很高的普及率，它使用 TypeScript 开发，提供了完整的类型系统；样式则是使用 Less 去实现。而且它的单元测试覆盖率超过 99%，因此功能非常稳定，这也是它能够得以普及的一个重要原因。

下图就显示了 AntD 提供的一个后台管理系统的模板：

AntD 的强大之处主要在于**提供了各种高级组件，可以满足企业级的复杂交互应用场景**。下面我对几个组件的特性进行一下简要的介绍，方便你有一个直观的了解：

1. Table：主要功能有虚拟滚动、列锁定、树状展开、内置的排序和过滤支持等等。这些功能对于企业级的大数据量的 UI 展现提供了很好的支持。
2. Tree：主要功能有动态加载节点数据、支持节点的拖拽、支持节点多选框、可内置于下拉框等等。
3. Form：提供了表单状态管理、布局系统、丰富的验证机制等功能。可以一站式满足表单的开发需求。

4. 消息和对话框：AntD 提供了非常易用的 API 用于一些快速的状态反馈。比如 Modal.success 用于显示一个弹出对话框；message.success 用于显示一个顶部的自动消失的反馈消息；notification.open 则用于显示一个通知栏消息，等等。

当然，这些只是整个 AntD UI 库的一小部分功能，但我们已经可以看到它面向企业级应用的设计原则，它将重点放在了提供更多开箱即用的高级组件上，以提高开发效率。那么更完整的 API，你可以去参考官方文档。

react-use

在 React Hooks 推出之后，来自 React 开发社区的 Vadim Dalecky 将一些常用的需求封装成了可重用的 Hooks，方便了 React 的开发。这个就是 **react-use 库**，提供了各种的工具 Hooks，了解并熟练使用它们，可以节省很多重复的工作量。

下面就一起来看看我认为最为常用的四个 Hooks 吧：

useSearchParams

其实在前面第 8 讲你已经看到了这个 Hook 的用法，就是用于获取 URL 中的查询字符串：

复制代码

```
1 const searchKey = useSearchParam("key")
```

这样，你就能获得 key 这个参数的值，并且在值发生变化时触发组件的刷新。

useEvent

如果用 DOM 的原生 API 去绑定事件，通常我们需要在组件创建时去监听，再在组件销毁时取消监听，比较繁琐。而 useEvent 这个 Hook 封装了这部分逻辑，让我们使用起来非常方便。

使用方法如下：

复制代码

```
useEvent('keydown', onKeyDown);
```

这样的话，我们就可以在页面上监听键盘按键的事件，而无需关心该在何时绑定监听，以及何时取消监听。

useCookie

Cookie 也是常用的数据源之一，但浏览器自带的 Cookie API 使用起来很不方便，需要自己去解析 Cookie 的值，而 useCookie 则可以让你很方便地去读取、更新或者删除某个 Cookie。

使用的方法如下：

复制代码

```
const [value, updateCookie, deleteCookie] = useCookie("my-cookie");
```

需要注意的是，这里的 Hook 的**作用域仅限于当前组件**。如果 Cookie 是在组件外部被修改，则是无法触发当前组件的刷新的。你可以把这个 Hook 的作用看作是跟 useState 类似的，只是 State 的值是持久化在 Cookie 里存储的。

usePrevious

这个 Hook 可以让你获取某个 State 的上一个值，有时我们需要比较前后值的变化，这时我们就能感受到 usePrevious 这个 Hook 的好用了。

使用的方法如下：

复制代码

```
const [count, setCount] = React.useState(0);
// 获取 cound 的上一个值
const prevCount = usePrevious(count);
```

小结

这节课我们主要介绍了 React 生态圈的四个常用工具库。主要的学习目的呢，就是为了让你对有助于 React 开发的重要项目做一个了解，从而避免在遇到一些需求时，自己费了很大力气去实现，却发现原来早有开源的方案。

这里尤其要强调的是 lodash。虽然它和 React 没有必然的关系，但是 lodash 非常擅长数据结构的各种转换，因此对 React 中的状态管理非常有帮助。那么熟悉它的 API，自然也是非常有必要的。

虽然这节课我只介绍了几个工具库，但是整个开源生态是非常庞大的，你在开发过程中也一定会去使用各种开源的项目。那么在这里我也和你分享一下开源项目选择的几个考虑因素。

1. 项目的作者是很认真地去提供开源产品，比如有完善的文档、充分的测试，以及及时的 Bug 修复速度。通常来说，只有这样才能做出可靠的产品。
2. 要全面了解选择的开源项目。如果你只是用到了一小部分内容，那么就要看它是否提供了模块化的机制，让你可以只使用其中的一部分，否则就会让你的项目越来越大。
3. 把 Github Star 和 npm 下载量作为参考指标：通常来说，越流行的项目也意味着更高的质量。如果一个项目开源了很久，而每周却只有几十个下载量，那你就要慎重考虑了。

总之呢，开源社区是一个大宝藏，有很多很好的项目等待着你去发掘。

思考题

文中主要分享了我个人一直使用的工具库，你有什么觉得用着非常顺手的工具库呢？和大家分享一下吧，说说它们给你带来的惊喜。

欢迎把你的想法和思考分享在留言区，我会和你交流讨论。也欢迎把课程分享给你的朋友和同事，大家共同进步。

分享给需要的人，Ta订阅后你可得 20 元现金奖励

赞 2 提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。 页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 18 | 单元测试：自定义 Hooks 应该如何进行单元测试？

下一篇 20 | React 的未来：什么是服务器端组件？

精选留言 (7)

写留言

Jerryz 置顶
2021-07-08

我来一个：classnames 用于拼接 class 的各种场景，用起来就是各种爽。

4

程
2021-07-09

react-use同类库，阿里的 ahooks

展开

4

2021-07-08

老师，可以讲一下react-query、swr这种库吗？可以完全替代redux吗

1 2

Harry

2021-07-08

这几个工具平常开发也在使用，怒赞👍

2

盖世英雄

2021-07-13

fecha处理时间日期

展开

作者回复: 类似的还有个 date-fns 。

渡

2021-07-09

王老师，我今天在尝试debounce时，引发了一个疑问，关于“FC组件会生成冗余函数对象”的问题。当时是想实现一个防抖的受控输入框，发现必需要借助useCallback才能真正实现防抖（原因很容易想明白：要防止输入框的onChange里每触发的回调里都是debounce新生成的函数。）

但这样就暴露了useCallback的一个问题：虽然useCallback返回的函数是不变的，但作...

展开

Geek_413aa8

2021-07-08

moment.js是一个 JavaScript 日期处理类库，用于解析、检验、操作、以及显示日期

1